International Journal of Geo-Information

MDPI

Article

Spatiotemporal Graph Convolutional Network for Multi-Scale Traffic Forecasting

Yi Wang and Changfeng Jing *

School of Geomatics and Urban Spatial Informatics, Beijing University of Civil Engineering and Architecture, Beijing 100044, China; 2108160119006@stu.bucea.edu.cn
* Correspondence: jingcf@bucea.edu.cn

Abstract: Benefiting from the rapid development of geospatial big data-related technologies, intelligent transportation systems (ITS) have become a part of people's daily life. Traffic volume forecasting is one of the indispensable tasks in ITS. The spatiotemporal graph neural network has attracted attention from academic and business domains for its powerful spatiotemporal pattern capturing capability. However, the existing work focused on the overall traffic network instead of traffic nodes, and the latter can be useful in learning different patterns among nodes. Moreover, there are few works that captured fine-grained node-specific spatiotemporal feature extraction at multiple scales at the same time. To unfold the node pattern, a node embedding parameter was designed to adaptively learn nodes patterns in adjacency matrix and graph convolution layer. To address this multi-scale problem, we adopted the idea of Res2Net and designed a hierarchical temporal attention layer and hierarchical adaptive graph convolution layer. Based on the above methods, a novel model, called Temporal Residual II Graph Convolutional Network (Tres2GCN), was proposed to capture not only multi-scale spatiotemporal but also fine-grained features. Tres2GCN was validated by comparing it with 10 baseline methods using two public traffic volume datasets. The results show that our model performs good accuracy, outperforming existing methods by up to 9.4%.

Keywords: traffic volume forecasting; fine-grained spatiotemporal feature; multiple-scale feature; spatiotemporal graph neural network

Citation: Wang, Y.; Jing, C. Spatiotemporal Graph Convolutional Network for Multi-Scale Traffic Forecasting. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 102. https://doi.org/10.3390/ijgi11020102

Academic Editors: Wolfgang Kainz and Marco Helbich

Received: 9 December 2021
Accepted: 29 January 2022
Published: 1 February 2022

1. Introduction

Geo-information-based Internet of Things (IoT) devices have become the infrastructure of the intelligent transportation system, which has advanced the rapid development of the intelligent transportation system. High accuracy short-term traffic forecasting, as the key mission to intelligent transportation systems (ITS), can effectively help road management, congestion relief, travel planning and many other applications. However, the nonlinearity and complexity of traffic flow make the spatiotemporal patterns of traffic flow dynamic, variable and difficult to understand. The temporal pattern refers to the dynamically changing traffic pattern, which shows periodicity and tendency; the spatial pattern refers to the interaction between nodes in a transportation network, which shows the traffic state at a point influenced by the upstream traffic condition of the connected road. Due to its great practical value, people have been working on more accurate forecasting models from the perspective of spatiotemporal patterns.

Existing traffic forecasting models can be divided into two types: statistical models and machine learning models. In the early traffic forecasting research, the traditional statistical traffic prediction models focus on temporal patterns. Among them, the representative models include AutoRegressive Integrated Moving Average (ARIMA) [1], Vector Auto-Regression (VAR) [2] and so on. These methods assume linearity of the data based on time series. However, the complexity and nonlinearity of traffic flow were unable to meet the ideal assumptions. The machine learning models, such as

Support Vector Regression (SVR) [3] and K-Nearest Neighbor (KNN) [4], are based on well-trained sample data to predict nonlinearity of the traffic flow, which means huge workload. Many traffic prediction models based on deep learning have been developed [5]. For example, convolutional neural networks (CNN) [6], recurrent neural networks (RNN) [7] (especially long short-term memory networks (LSTM) [8] and gated recurrent unit networks (GRU) [9]) were engaged to predict traffic flow or speed. However, it is not sufficient to extract temporal features alone because there is also spatial dependence in the traffic data. In recent years, many studies have constructed road networks as graph structures and based on graph neural networks (GNN) [10] for spatial feature extraction. After that, they combine CNN or RNN approaches with them to construct model, i.e., Spatiotemporal Graph Neural Networks (STGNNs) [11,12], to capture spatiotemporal features. Among them, CNN-based models [11,12] are represented by STGCN, MSTGCN, ASTGCN, Graph WaveNet, etc., while RNN-based models [11,12] are represented by GCRNN, DCRNN, T-GCN, AGCRN, etc. Compared with the traditional deep learning methods, they take more spatial features into consideration, which improve the prediction accuracy, and also become the mainstream methods for traffic forecasting at present.

In recent years, the existing STGNNs has been gradually developed, but nevertheless, they can also be optimized again in terms of feature extraction. The specific questions are as follows: (1) Fine grain. Due to the GNN construction, they intend to obtain the shared patterns of nodes in traffic sequences and ignore the pattern differences between nodes, resulting in the inability to obtain fine-grained spatiotemporal features between nodes. (2) Multi-scale. STGNNs are mostly used for feature extraction at a single scale. However, the spatiotemporal patterns at a single scale have some limitations, and they make the receptive field fixed in the spatial range. The spatiotemporal information extracted at different spatiotemporal scales is different but interconnected, which affects the prediction results. (3) Connection method. The way the spatiotemporal features are connected also affects the transmission of information, which eventually affects the results. In order not to let the information be lost in the transmission, we need to strengthen the feature transmission process and widen the width of features as much as possible. At present, there is hardly a method that considers all of these problems, which also limits the performance of its models.

To address the aforementioned problems, we proposed a novel spatiotemporal graph neural network framework—Temporal Residual II Graph Convolution Network (TRes2GCN)—to optimize the extraction of capturing spatiotemporal features, ultimately improving the prediction accuracy. The contributions of the proposed TRes2GCN are as follows:

- To optimize the extraction of a feature, a novel spatiotemporal graph neural network model was proposed that simultaneously considers temporal periodicity, spatiotemporal multi-scale features, connection method, and node pattern embedding.
- Based on Res2Net, we design hierarchical temporal attention layers and hierarchical adaptive graph convolution, so as to learn multi-scale spatiotemporal features. To the best of our knowledge, this paper is the first study to apply the idea of Res2Net in the field of spatiotemporal graph neural networks for traffic forecasting.
- Systematic experiments were conducted to compare our approach with existing state-of-the-art methods using two publicly available real-world traffic volume datasets. The results show that our model performs good accuracy, outperforming existing methods by up to 9.4%.

The remainder of this paper is organized as follows: Section 2 reviews the literature related to traffic flow forecasting using the STGNNs method. In Section 3, we present related knowledge used in the following sections as a basis, firstly. Then, we give the design and detail of our proposed model. Section 4 details the experiment, baseline models in this work and results. Section 5 discusses and further analyzes the new model and results. Section 6 summarizes our work.

2. Related Work

2.1. Short-Term Traffic Volume Forecasting Models

Short-term traffic forecasting research produced many methods and can be generally divided into two categories, statistical methods and machine learning methods. Many of the statistical methods are linear models focusing on interpretability and requiring a more solid hypothesis. The representative works include HA [13], ARIMA [1], VAR [2], partial least squares (PLS) [14], etc. However, these models do not perform well in practical scenario applications because they do not consider the dynamic and nonlinear characteristics of traffic data and have difficulty satisfying their stationary linear assumptions. Machine learning, on the other hand, can learn more complex and effective spatiotemporal correlations from the data itself, thus providing better prediction results. Some of the representative works are SVR [3], KNN [4] and neural networks [15]. Although the features they extract contain nonlinear information, they are more time-consuming and uncertain because their features are mostly extracted manually by humans.

With the development of deep learning, they are able to automatically model more complex dependencies, which has drawn great attention to modeling complex spatiotemporal data [5]. Many classical deep learning models have been used for traffic prediction due to their natural sequence processing capabilities, e.g., RNN [7], LSTM [8] and GRU [9]. Subsequently, others have been used to model traffic time series by CNN, learning temporal information with convolutional kernels and verifying their superiority over other recurrent neural networks [6]. However, these methods tend to focus only on the temporal patterns of traffic data and ignore the spatial patterns. To fully learn the spatial patterns, scholars divide the traffic scenes into image-model scenes for learning. CLTFP [16] tries to capture the spatial features using CNN and build a spatiotemporal prediction network using LSTM to solve the short-term temporal prediction problem. Convolutional LSTM (Conv-LSTM) [17] also use CNN and LSTM networks to capture spatial and temporal features, respectively. However, the difference is that instead of simply stacking a CNN and an LSTM network, the Conv-LSTM network places the CNN in the gating of the LSTM network and merges them intrinsically. ST-ResNet [18] uses deep residual CNN to manage the intensity on a spatial grid to predict citywide crowd flows at different time spans. They also input spatiotemporal features for different time periods and add external factors to improve the accuracy of the predictions. All the above methods are processed for grid data, while traffic flow is essentially a graphical data. It needs to be converted into a format that is not convenient enough. In addition, they focus only on the neighborhood information of the grid data, ignoring certain topological information implied by the interconnected road segments.

2.2. Spatiotemporal Graph Neural Network for Traffic Forecasting

Compared with convolutional neural networks in Euclidean space, graph neural networks (GNN) can sample and aggregate under disordered and irregular spaces and are more suitable for processing graph structured data. With the increasing popularity of GNN, several approaches to GNN have emerged in recent years, such as graph convolutional networks (GCN) [19], Chebyshev networks (ChebNet) [20], graph attention networks (GAT) [21], diffusion convolutional neural networks (DCNN) [22], etc. Especially, GCN are particularly popular. They are widely used in graph structure classification, recommendation systems, etc. Since the spatial characteristics of traffic data match well with the graph structure, they have also become key to obtaining the intrinsic spatial features of the data. In recent years, their continuous development has made STGNNs the mainstream model in traffic prediction models.

Although there are many variants of STGNN, overall, they can be divided into two kinds: one is the RNN-based model and the other is the CNN-based model.

One of the RNN-based STGNN models is used to capture temporal features with a recurrent neural network and replace (or directly add) the linear layer of the RNN with a graph convolution to capture spatial features. For example, regarding the most representa-

tive T-GCN [23] or GCRNN [24], both of them use GRU to capture temporal features and GCN to capture spatial features, thus allowing them to capture the complete spatiotemporal features. Due to the drawback of convolutional kernel sharing and sharing of GCN, other GNN are considered for learning. Li proposed the classical model DCRNN [24], which captures spatial features by random wandering of DCNN on the graph and captures temporal features using Seq2Seq, making the model more flexible and efficient. However, traffic flow is dynamic and changing data, and it needs to be considered that the prediction may vary at each step. Therefore, Cui proposed TGC-LSTM [25], which integrates GCN and LSTM and optimizes the workflow of graph convolution by adding a free-flow reachability matrix. Guo developed OGCRNN [26], which optimizes on the basis of GCRNN. It uses the data variation itself to optimize the Laplace matrix during graph convolution, thus enabling the model to obtain dynamic spatiotemporal features. Both of these works improve the fixed spatiotemporal correlations, thus enabling dynamic changes in spatiotemporal correlations. Bai proposed A3T-GCN [27], which adds attention to T-GCN and considers the dynamic changes of data in the feature acquisition phase. However, RNN has limitations, since it shares the same parameters at each time step, which denotes a weak ability to describe the complex dynamics of temporal correlations.

Many STGNN models based on CNN were developed. These methods often capture temporal feature with CNN and spatial feature with GNN. The most representative one is STGCN [28], which captures temporal features with CNN with GLU gating and spatial features with GCN. It is the same as T-GCN to obtain the complete spatiotemporal features. However, due to the very small range of the convolution kernel of CNN, it has a relatively small perceptual field, which also leads to its reduced long-term prediction ability. To solve this problem, Graph WaveNet [29], proposed by Wu, want to improve the accuracy of long sequence prediction by using dilation convolution. It also creates an autonomous learning adjacency matrix to help learn the unknown adjacency matrix, which is further adaptive on a dynamic basis. This is the first study on adaptive adjacency matrix in the field of traffic prediction. In addition, there are other models that consider the attention mechanism to compensate the problem of a small perceptual field. Guo proposed ASTGCN [30] to integrate the attention mechanism and GCN to delineate the dynamic spatiotemporal correlation. It not only expands the receptive field through attention, but also cleverly uses temporal and spatial attention mechanisms to achieve the ability to obtain dynamic spatiotemporal information entirely from the data itself. They also propose the MSTGCN [30] without dynamic attention for comparison and demonstrate that his ASTGCN can capture dynamic spatiotemporal features. Zheng proposed GMAN [31], which also focuses on the dynamic changes in the process. Compared with ASTGCN, GMAN completely replaces CNN and GCN with attention and is more concerned with the interaction between temporal and spatial information. GMAN is designed based on autoencoder structure and attention mechanism to calculate the effect of spatial and temporal factors on traffic conditions. Later, some models based on ASTGCN and GMAN were developed and operated with some improvements, such as DGCN [32] and MASTGCN [33]. Due to their powerful performance, such models are also used for trajectory prediction [11] and traffic data imputation [11,12,34].

There are also studies that want to add other data (weather conditions, points of interest (POI), etc.) to the above studies to aid in making predictions. Bai proposes a cascaded graph convolutional recurrent neural network [35] to model and predict historical vehicle demand data in the city. It embeds meteorological data and temporal elements in an encoded manner and forecasts them by means of an LSTM-based Auto-encoder. Zhu proposes a prediction model, called AST-GCN [36], that considers both POI and weather effects on traffic flow. It fuses the three kinds of data and puts them into T-GCN for training, which is simple and effective. Although the above methods are very effective, not all data are fully accessible due to data confidentiality and privacy issues, which limit the work of their models.

All the above models basically perform feature extraction at the same spatiotemporal scale. However, there are some differences in the spatiotemporal patterns being learned due to the different spatiotemporal scales. Therefore, some scholars have studied local spatiotemporal features or multi-scale features. Yu proposed ST-UNet [37], which can obtain multi-scale spatial information. ST-UNet is based on the model framework of U-Net structure, and jointly uses ST-pooling and ST-unpooling to accomplish the extraction of spatial information at different scales. In addition, it uses expanded RNN to obtain longer temporal features. Song proposed STSGCN [38], which considers feature extraction of local graphs and proves the significance of local graph features. Based on GCN, it constructs spatiotemporal blocks that can perform both temporal and spatial information extraction, which also considers the heterogeneity of traffic flow prediction. Guo proposed HGCN [39], which hierarchically processes the graph structure based on spectral clustering to accomplish traffic prediction on both micro and macro graph structures. This superposition of microscopic and macroscopic features also achieves multi-scale spatiotemporal feature capture. However, the scale size of the proposed multi-scale models needs to be adjusted artificially. Due to the error of manual adjustment, it limits the potential capability of the model on the one hand; on the other hand, it requires more manual operations, which is very laborious and troublesome.

The current studies, however, have studied the problems in graph structure at the granularity of graphs or local graphs, and there is a lack of node-specific fine-grained studies. Bai proposed AGCRN [40] to study the prediction in detail with node granularity. It uses a GCN with node parameters to generate adaptive adjacency matrix and embeds the parameters with GRU to learn traffic patterns dynamically. It solves part of the problems of RNN with GCN and improves the ability of some models to describe the complex dynamics of fine-grained temporal and spatial correlation. However, it still has some shortcomings, ignoring that there will be some differences in the patterns of nodes at different scales. On the other hand, even the parameter sharing problem of RNN is alleviated with node embedding. However, certain training is stopped, the node embedding information is fixed and the parameter features of the RNN, though much richer than the original, are still fixed. Therefore, when it is swapped to another part of the data for testing, its model performance will still be limited.

3. Methodology

3.1. Preliminary

Based on the concept of the graph construction, the traffic network can be defined as $G(V, E, A)$, where $V \in R^N$ denotes a set of vertices or nodes and $E \in R^{N \times N}$ represents a set of edges. The traffic network G represents the relationship between the vertices in the spatial dimension and remains constant over time. The adjacency matrix is represented by $A \in R^{N \times N}$. The graph signal matrix is denoted as $X_t^G \in R^{N \times F}$, where F represents the features of each node at time t. In terms of the traffic flow forecasting, each time t has a feature matrix X_t, the historical data H can be denoted as $X_H = \{X_{t-i+1}, X_{t-i+2}, \ldots, X_t\}$, and the predicted data P can be denoted as $X_P = \{X_{t+1}, X_{t+2}, \ldots, X_{t+p}\}$.

3.2. Design of TRes2GCN

In this section, Figure 1a shows the proposed TRes2GCN framework. The model is composed by three components, each of which contains the same Temporal Residual II Graph Convolutional Submodule (TRes2GC-Submodule) part (Figure 1b). Three Submodules, respectively, focus on three different time periods, and the weighted fusion of each time period is performed by a set of fusion parameters. Each TRes2GC-Submodule contains four spatiotemporal blocks (ST Blocks), which are connected in Dense connection [41].

In summary, our model has several design implications, as follows:

1. Through a multi-component approach, our model learns and fuses spatiotemporal features of different time periods and explores the travel patterns of vehicles.

2. TRes2GC-Submodule connects spatiotemporal blocks in the way of DenseNet, which increases the width of features by fusing spatiotemporal features of different levels and effectively mitigates the problem of network degradation and oversmoothing.

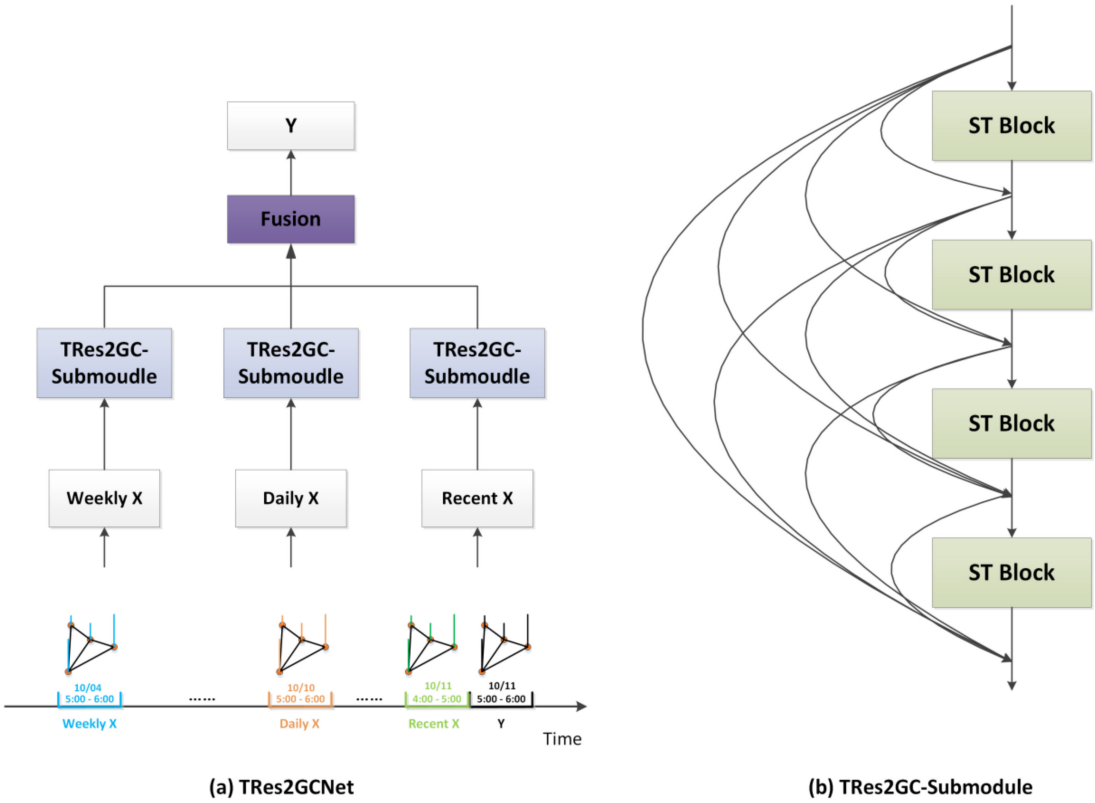

Figure 1. The framework of (**a**) TRes2GCN and (**b**) TRes2GC-Submodule.

3.3. Multiple Temporal Periods

Although traffic data are characterized by real-time dynamic changes, studies have shown that traffic data have different degrees of similar patterns in proximity, daily and weekly periods [17]. Considering this periodic variation of traffic flow can effectively improve the accuracy of prediction. In this study, we used ASTGCN [30] to sample the time period.

Assume that the current time is T_0, the forecast window size is T_p and the number of samples per day is q. We intercept three time series T_h, T_d and Tw, respectively, on the time axis as the time component inputs for the recent periods, daily periods and weekly periods, where T_h, T_d and T_w are of the same quantity as T_p. We have demonstrated the different time periods for the input. The details of these three periods are described as follows:

(1) Recent period: Recent period refers to the historical data in the nearby of the forecast value, denoted as $X_h = \left(X_{T_0-T_k+1}, X_{T_0-T_k+2}, \ldots, X_{T_0}\right) \in R^{N\times F\times T_k}$ Since sudden changes in traffic flow are precursory, the near moment fragment is particularly important for the forecast fragment. The specific slice is shown in Figure 1a, and the green color relative to the black color is its recent period.

(2) Daily period: A daily period refers to the historical data of one day ago at the same time as the forecast segment, denoted as $X_d = \left(X_{T_0-q+1}, X_{T_0-q+2}, \ldots, X_{T_0-q+T_p}\right) \in R^{N\times F\times T_d}$ It is a fragment of the same time interval as the forecast period in the past day. The traffic data are likely to show a part of the same pattern over some time, for example, there are morning peak and evening peak for each day of a weekday. Therefore, we choose this segment as part of the common forecast, thus capturing the similar

characteristics of the daily period. The specific slice is shown in Figure 1a, and the orange color relative to the black color is its daily period.

(3) Weekly period: A weekly period refers to the historical data of a week ago at the same time as the forecast segment, denoted as $X_w = \left(X_{T_0-7*q+1}, X_{T_0-7*q+2}, \ldots, X_{T_0-7*q+T_p}\right) \in R^{N\times F\times T_w}$ It is a fragment of the same time interval as the forecast period in the past week. The reason is the same as the daily period. For example, the flow change of this Friday is very similar to next Friday, but there are some differences with the flow change of the weekend. Thus, we use it to capture the similar characteristics of the weekly period. The specific slice is shown in Figure 1a, and the blue color relative to the black color is its weekly period.

3.4. Spatiotemporal Feature Capture Method

Each spatiotemporal block (Figure 2a) contains a hierarchical temporal attention layer (Figure 2b) and a hierarchical adaptive graph convolution layer (Figure 2c). They are not in the usual vertical arrangement, but are deployed horizontally. They interact with each other under a gating influence to make the spatiotemporal features more closely related. As for the hierarchical attention layer and the hierarchical adaptive graph convolution layer, they implement the capture of multi-scale temporal and multi-scale spatial information on a layer-by-layer basis, respectively. This is combined with the block-based multi-scale information capture mentioned above, thus enabling our framework to accomplish multi-scale spatiotemporal information capture at fine granularity. The problem of single spatiotemporal feature in non-Euclidean data prediction problem is fundamentally solved.

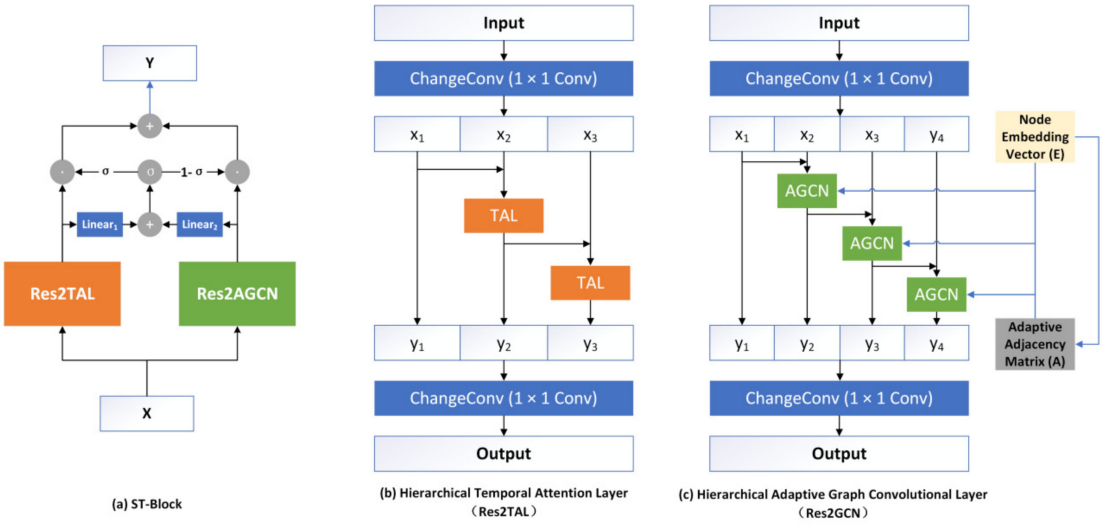

Figure 2. The (**a**) ST Block of TRes2GCN. The framework of (**b**) hierarchical temporal attention layer and (**c**) hierarchical adaptive graph convolution layer.

(1). Hierarchical Temporal Attention Layer

Temporal Attention Layer (TAL). For the traffic flow, it is dynamic and relevant in the time dimension, and such dynamic features cannot be learned by ordinary CNN or RNN. Therefore, this paper uses the attention mechanism in the NLP domain [42], thus focusing on the importance of information at different time nodes in a complete period of time and assigning greater weights to valid time points [30]. It not only adds dynamic and adaptive temporal relevance to our model, but also expands the feeling field to solve longer time prediction problems. Its mathematical formula is shown as follows:

$$E = V_e \cdot \sigma\left(\left(\left(X_h\right)^T U_1\right) U_2 (U_3 X_h) + b_e\right), \quad (1)$$

$$E'_{i,j} = SoftMax(E_{i,j}) = \frac{exp(E_{i,j})}{\sum_{j=1}^{T} exp(E_{i,j})}, \quad (2)$$

$$TAL(X_h) = X_h E' = (X_1, X_2, \ldots, X_T) \cdot E', \quad (3)$$

where $V_e, b_e \in R^{T \times T}$, $U_1 \in R^N$, $U_2 \in R^{F \times N}$, and $U_3 \in R^F$ are learnable parameters, $X_h = (X_1, X_2, \ldots, X_T) \in R^{N \times F \times T}$ represents the output of the previous spatiotemporal block, E denotes the attention weight of each moment relative to other moments, and it will vary depending on the input data and the value of $E_{i,j}$ indicates the correlation between time i and time j after normalization (the $SoftMax$ activation function is used for normalization).

Hierarchical temporal attention layer (Res2TAL). Res2Net [43] has proven its ability in the field of computer vision (Euclidean convolution) to divide a single convolution into multiple convolutions that are interrelated, allowing us to extract multi-scale feature information from a fine-grained perspective. We built on the temporal attention layer by deploying the framework part of Res2Net with the residual structure removed, thus proposing a hierarchical temporal attention layer, as shown in Figure 2b. The hierarchical temporal attention layer not only focuses on the importance of each moment, but also learns the similarity of temporal trends over a period of time. This allows us to learn both the temporal value weight changes and temporal trend changes over a period of time, thus learning temporal features in a more fine-grained perspective. Res2Net, on the other hand, can be used without increasing the number of parameters with more training time under specific settings. Therefore, Res2TAL can enable us to extract temporal feature information at different scales without increasing parameters. Its mathematical formula is shown as follows:

$$[Z_{h1}, Z_{h2}, \ldots, Z_{hs}] = Z_h = ChangeConv_1(X_h), \quad (4)$$

$$Y_{hi} = \begin{cases} Z_h, & i = 1; \\ TAL_i(Z_{hi} + Y_{hi-1}), & 2 \ll i \ll s. \end{cases}, \quad (5)$$

$$Res2TAL(X_h) = ChangeConv_2([Y_{h1}, Y_{h2}, \ldots, Y_{hs}]), \quad (6)$$

where s is the parameter of scale in Res2Net and also represents the number of our time scales. Both $ChangeConv_1$ and $ChangeConv_2$ are convolutional neural networks with a convolutional kernel of 1, used to deflate the number of channels of the features.

It is worth mentioning that we removed its squeeze and excitation block (SE-Block), which originally existed [44] to replace the blending between single layers, and we blended multi-scale temporal and multi-scale spatial in the form of spatiotemporal blocks. Instead of extracting shared features, we want to retain the inter-scale difference. This facilitates the interactions between multi-scale information in spatiotemporal block gating, while the small improvement of its performance is verified later.

(2). Hierarchical Adaptive Graph Convolutional Layer

Adaptive adjacency matrix generation. In the spatial dimension, there are differences in traffic patterns at each node, which can have a huge impact on traffic prediction. The current way of graph construction mostly starts from individual attributes (e.g., road network, point of interest (POI), traffic similarity, etc.), which are highly interpretable but do not contain the complete spatial dependency information. In addition, most graph structures require predefined graphs in advance, and their models cannot work when they are missing. In order to solve the above appeared problems, we applied a node embedding approach to the adaptive adjacency matrix construction [29,40]. The formula is shown as follows:

$$A_{adp} = SoftMax(ReLU(E \cdot E^T)), \quad (7)$$

where $E \in R^{N \times D}$ denotes the node embedding parameter in D dimension, which can be learned adaptively and used to learn the variability among nodes, $I_N \in R^{N \times N}$ represents the unit matrix of N dimensions, which is used to add self-loops to the graph structure, the

ReLU activation function is used to eliminate negative connections between nodes and the SoftMax activation function is used to normalize the adjacency matrix.

Adaptive Graph Convolution Layer (AGCN). Since the convolution kernel of GCN is shared, although it can capture the most important traffic patterns in the whole traffic graph, it is actually difficult for us to learn the variability among nodes and learn the traffic patterns of different nodes in a fine-grained manner. AGCN [40] enables us to accomplish fine-grained feature capture of nodes without the need of known node attribute data. It adds one more node embedding parameter in the above adjacency matrix to the GCN. This parameter can help us train the adaptive adjacency matrix while influencing the learning parameters during graph convolution and giving a different deflation of the features of each node. Therefore, it helps us to consider the node pattern differences so as to obtain dynamic adaptive spatial features at fine-granularity. Its equation is shown as follows:

$$AGCN(X) = A_{adp}XEW + Eb, \tag{8}$$

where $W \in R^{D\times F\times F'}$ and $b \in R^{D\times F'}$ are learnable parameters. Different from GCN, it utilizes the idea of matrix decomposition to solve part of the overfitting and oversmoothing problems of GCN. However, it also has a problem in that it has a fixed range of perceptual fields. This limits the multi-scale spatial features captured by the model.

Hierarchical Adaptive Graph Convolution Layer (Res2GCN). Res2Net [43] has been verified to have a better ability in extracting features at multiple scales in Euclidean space. We refer to this idea and deploy the construction idea of Res2Net to AGCN to propose a hierarchical adaptive graph convolution layer. Similarly, it is able to capture multi-scale spatial features in traffic flow (non-Euclidean space) at a finer granularity level, and the internal operation is shown in Figure 2c. The multi-scale features of Res2GCN not only increase the perceptual field of convolution, but also alleviate the performance limitations caused by the GCN kernel fixation as well as sharing problems to a certain extent, because the AGCN within each scale are not sharing parameters. On the other hand, since the same embedding vector is used by AGCN at different scales in the parameter back propagation process, this allows our adjacency matrix's to also fuse to learn node patterns at multiple scales. In a later section, we also verify that different adjacency matrices with node embedding parameter approaches have different effects on Res2GCN. The equation is described as follows:

$$[Z_1, Z_2, \ldots, Z_s] = Z = ChangeConv_1(X), \tag{9}$$

$$Y_i = \begin{cases} Z_i, & i = 1; \\ AGCN_i(Z_i + Y_{i-1}), & 2 \ll i \ll s. \end{cases}, \tag{10}$$

$$Res2AGCN(X) = ChangeConv_2([Y_1, Y_2, \ldots, Y_s]), \tag{11}$$

$$Res2AGCN(X_h) = [Res2AGCN(X_1), Res2AGCN(X_2), \ldots, Res2AGCN(X_T)], \tag{12}$$

where s is the parameter of scale in Res2Net and also represents the number of our spatial scales and $ChangeConv_1$ and $ChangeConv_2$ are a convolutional neural network with a convolutional kernel of 1, playing the same role as in Res2TAL.

(3). Dynamic Gated Fusion

This section is used to explain how the temporal and spatial features within the ST Block are integrated. The traffic flow forecasting of any road section needs to focus on both the historical flow of that road section and the historical flow of other related road sections in the past. Therefore, we need to dynamically fuse the extracted hierarchical temporal features with the hierarchical spatial features here. In the past studies, most of the models operate sequentially in a longitudinal direction, which will make the correlation between temporal and spatial features not strong enough. Therefore, we use a gating technique to fuse temporal and spatial features horizontally and dynamically, thus enhancing the

correlation between temporal and spatial features. The intrinsic operation process is shown in Figure 2a, as well as the mathematical equation shown below:

$$Gate = \sigma(Linear_1(Res2TAL(X)) + Linear_2(Res2GCN(X))), \tag{13}$$

$$ST\ Block(X) = Gate \odot Res2TAL(X) + (1 - Gate) \odot Res2GCN(X), \tag{14}$$

where $Linear_1, Linear_2$ is the linear transform layer, which can change dynamically and adaptively with the data itself and influence the magnitude of the action of $Gate$, $\odot$ is the Hadamard product and σ is the sigmoid activation function, which can be used in normalizing the feature values and, thus, avoiding gradient explosion during the fusion process.

3.5. Dense Connection

Most of the past studies, such as STGCN [28], ASTGCN [30] etc., have used residual connections for feature integration, and the continuous use of the same function in residual connections still hinders feature transfer. Since the TRes2GC-Submodule contains multiple ST blocks, the continued use of the ResNet structure [45] can cause problems such as gradient disappearance and network degradation. To solve these problems, we use DenseNet [41] for connecting our ST blocks. It enhances the propagation of features, encourages feature reuse and reduces the number of parameters. On the other hand, DenseNet can further alleviate the oversmoothing problem caused by multiple uses of GCN.

Fortunately, traffic flow data are not as complex as image data, which require a large number of layers to deepen the learning. It needs a reasonable and limited number of blocks, but not an extraordinary large number of blocks, to complete the learning of features. Therefore, it is worth emphasizing that our primary reason for using DenseNet is to widen the number of channels of spatiotemporal features, and the secondary reason is to perform network deepening. In the later ablation experiments, we also verify the differences brought by the different connection methods (and the differences between ResNet and DenseNet) and the effects brought by model widening and deepening on the prediction. The intrinsic operation is shown in Figure 1b, as well as the following equation:

$$\widetilde{Y}_{Sub} = ST\ Block_l([X^1, X^2, \ldots, X^{l-1}]), \tag{15}$$

where $ST\ Block_l$ represents the ST block of the block of the stack and $[X^1, X^2, \ldots, X^{l-1}]$ refers to the connection of the feature mapping generated after the passage of the $(0, \ldots, l-1)$ blocks.

3.6. Multi-Component Fusion

This section describes how to fuse multiple TRes2GC-Submodules. Since a multi-component format was used to our model in different time periods and different locations, we needed a way to fuse the features in the different components. For example, the Sunday model has little effect on the flow forecast of the Monday model, but the Monday model has a large effect on the Tuesday model. Therefore, we defined three weight parameters to self-adaptively deflate the importance of each component and summed them. Their formulas are shown below:

$$\widetilde{Y} = W_h \odot \widetilde{Y}_h + W_d \odot \widetilde{Y}_d + W_w \odot \widetilde{Y}_w, \tag{16}$$

where $\odot$ is the Hadamard product and W_h, W_d, and $W_w \in R^{N \times T}$ are the parameters to be learned and also the weight parameters for different cycles, respectively.

4. Experiment and Result

4.1. Datasets

Two public real-world traffic volume datasets were selected: PeMS04 and PeMS08, which were collected by the Caltrans Performance Measurement System in major urban areas of California [46]. All data are collected from over 39,000 sensors deployed on California freeways at 30 s intervals.

The raw data need to be preprocessed before it can be put into the model for training. We use the same preprocessing approach as STSGCN [38] and AGCRN [40] for some missing values in the dataset, we fill in these missing values by linear interpolation. In addition, we normalized the dataset with all the data values between [0,1] and the mean value of the dataset is 0. We divided the dataset chronologically and divided it into training, validation and test sets in the ratio of 6:2:2. The data were eventually aggregated into 5-min intervals, resulting in a total of 288 data points per day. The metadata information is shown in Table 1. Where the data average represents the mean of every five minutes (per time period) of data, and the data range represents the upper and lower limits of data variation every five minutes.

Table 1. Details of the datasets.

Dataset	Nodes (Sensors)	Edges	Time Steps	Time Range	Data Range (Per Time Period)	Average (Per Time Period)
PeMS04	307	341	16,992	1/1/2018–2/28/2018	0–919	91.74
PeMS08	170	295	17,856	7/1/2016–8/31/2016	0–1147	98.17

4.2. Baseline Methods

We choose the following 10 traffic flow forecasting methods as baseline models to compare with our model. Additionally, to demonstrate the novelty and validity of our model, seven of the baselines are state-of-the-art traffic flow forecasting methods. The details are as follows:

- **VAR**: Vector Auto-Regression is a forecasting model that captures the spatiotemporal feature between traffic data.
- **SVR**: Support Vector Regression utilizes a support vector machine to perform linear regression.
- **LSTM**: The long short-term memory network is a variant model of RNN that can better handle time-series tasks.
- **DCRNN**: The diffusion convolution recurrent neural network is an auto-encoder framework. It uses diffusion map convolution to obtain spatial features and Seq2Seq to encode temporal information.
- **STGCN**: The spatiotemporal graph convolutional network uses ChebNet to obtain spatial correlation and CNN with a gating mechanism to obtain temporal correlation.
- **MSTGCN**: The multi-component spatiotemporal graph convolution network extracts and fuses spatiotemporal information in different time periods by modeling different temporal patterns. It obtains temporal features by CNN and spatial features by ChebNet.
- **ASTGCN**: The attention-based spatiotemporal graph convolutional network adds on temporal attention and spatial attention to MSTGCN to extract dynamic spatiotemporal information.
- **Graph WaveNet**: Graph WaveNet combines GCN and dilated convolution network to obtain spatial correlation and temporal correlation separately. It also utilizes node embedding to adaptively learn adjacency matrix from the data.

- **STSGCN**: The spatiotemporal synchronous graph convolutional networks employ GCN to construct spatiotemporal synchronous convolutional blocks to synchronously obtain temporal and spatial correlations by stacking spatiotemporal synchronous convolutional modules.
- **AGCRN**: The adaptive graph convolutional recurrent network proposed a novel adaptive graph convolutional network so as to capture fine-grained spatial feature. In addition, it employs amplified GRU to capture the temporal feature.

4.3. Experiment Settings

TRes2GCN is implemented based on the PyTorch framework, and its training is performed on an NVIDIA RTX 2080TI graphics card. In our model, the 4-scale 26-width is taken for Res2GCN deployment and the 3-scale 26-width for Res2TAL. In addition to this, each other layer has 64 features. The batchsize of this model is 32, and the Adam optimizer is used for 40 epochs of model training with a learning rate of 0.001. In the training process, we decay the learning rate by cosine annealing and set the lowest point of learning rate decay to 0.0001, which facilitates us to get better training results. In the loss function selection, we use SmoothL1Loss (i.e., HuberLoss [47]) instead of L2Loss, so as to increase the anti-disturbance of the model and reduce the error caused by noise. The mathematical formula is as follows:

$$Smooth_{L1}(Y, \widetilde{Y}) = \begin{cases} 0.5(Y-\widetilde{Y})^2, & |Y-\widetilde{Y}| \leq \sigma; \\ \sigma|Y-\widetilde{Y}| - 0.5\sigma^2, & otherwise. \end{cases}, \tag{17}$$

where Y denotes the true value of traffic flow, $\widetilde{Y}$ denotes the value of traffic flow predicted by the model, and σ is a threshold parameter that controls the loss range of the squared error.

The experiment also deploys three widely used evaluation metrics to measure model performance, namely mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE), same as AGCRN [40]. We use these three metrics to measure the predictive effectiveness of our model compared to the baseline model.

4.4. Results Comparison

Table 2 gives a comparison of the average performance of the different frameworks over all time horizons. Overall, in both datasets, our model outperforms the other ten baseline methods in all three metrics. To describe clearly the predictive power of the proposed model, we analyze it in detail from two aspects.

Table 2. Performance comparison of different traffic flow forecasting methods.

Baseline Methods												
Datasets	**Evaluation Metrics**	**VAR**	**SVR**	**LSTM**	**DCRNN**	**STGCN**	**MSTGCN**	**ASTGCN**	**Graph WaveNet**	**STSGCN**	**AGCRN**	**TRes2GCN (ours)**
PeMS04	RMSE	36.66	44.59	40.74	37.12	37.07	35.48	34.50	39.70	33.83	32.26	**31.98**
	MAE	23.75	28.66	26.81	23.65	24.43	22.65	21.97	25.45	21.08	19.83	**19.62**
	MAPE (%)	18.09	19.15	22.33	16.05	18.34	16.32	15.47	17.29	13.88	12.97	**12.96**
PeMS08	RMSE	33.83	36.15	33.59	28.29	30.11	28.27	26.91	31.05	26.83	25.22	**24.52**
	MAE	22.32	23.25	22.19	18.22	19.95	18.54	17.37	19.13	17.10	15.95	**14.45**
	MAPE (%)	14.47	14.71	18.74	11.56	14.27	13.04	12.28	12.68	10.90	10.09	**9.50**

(1) Overall forecasting ability

In Table 2, we can see that the SVR and LSTM methods are less accurate compared to other networks because they do not take into account spatial correlation in the traffic data. VAR has better performance compared with SVR because some spatial information is considered. However, these methods have room for improvement in handling spatial information compared to spatiotemporal graph neural network methods, which is the reason why VAR performs mostly inferior to them.

STGCN, DCRNN, MSTGCN, ASTGCN, Graph WaveNet, STSGCN and AGCRN all consider complete global spatial correlation and have better performance compared to the above models. However, due to the different design and construction of each model, these models still show disparity in results. STGCN, DCRNN and Graph WaveNet all target only single-period feature capture from a design perspective, ignoring the importance of temporal cycles, and thus, will have slightly inferior performance. For MSTGCN and ASTGCN, they consider different time periods and will tap more time period information than STGCN and DCRNN like this. However, the above models lack multi-scale temporal information compared to the proposed TRes2GCN, thus limiting their performance. For STSGCN, local graph features are also considered, thus providing another partial improvement compared to STGCN, DCRNN or Graph WaveNet. However, it has a single extraction method, so the extracted spatiotemporal features are not rich enough. In contrast, our model incorporates spatiotemporal features of different levels and scales through the DenseNet and Res2Net structures, which are richer in spatiotemporal features and have better prediction results. For AGCRN, it cracks the learning parameters in GCN into two small parameters for learning, which is improved from a fine-grained perspective. However, it only considers the fine-grained spatial and temporal features at a single scale, and it is not able to consider the travel patterns at different time periods. Our proposed model, on the other hand, deploys AGCN and TAL through the idea of Res2Net based on the cleavage of GCN, and completes the capture of finer-grained multi-scale spatiotemporal information. On top of this, we also achieve feature capture at different time periods through a multi-component model to accomplish higher accuracy prediction.

(2) Multi-step prediction capability

To explore the performance of our model versus the baseline model over different time spans, we visualize the results of the metrics at different time steps (Figure 3). It is clear that for all models, the prediction errors show an increasing trend as the prediction time increases. Among them, SVR with LSTM has very good short-term prediction, but its error has the fastest upward trend among all models, because it fails to consider the change of spatial characteristics. Compared with VAR, its short-term prediction ability is not very good. However, because it takes into account some spatial features, the rising trend of its error has slowed down more with time. This also proves that the consideration of spatial features can effectively improve the prediction ability of the model in the medium and long term. As for the spatiotemporal graph neural networks, in most cases, their medium- and long-term predictions are better than VAR, SVR and LSTM. This also demonstrates the effectiveness of GCN in capturing spatial features in traffic prediction. It is worth mentioning that AGCRN and our TRes2GCN, which consider fine-grained spatial features, are significantly due to other baseline models in the medium- and long-term prediction task. This also highlights the effectiveness of AGCN compared to the normal GCN. TRes2GCN has a large improvement over AGCRN, especially for short- and medium-term forecasting. This is due to the advantage of considering different time periods, which allows our model to be known to have more characteristics of the changes in this time period, thus fitting our prediction results more closely.

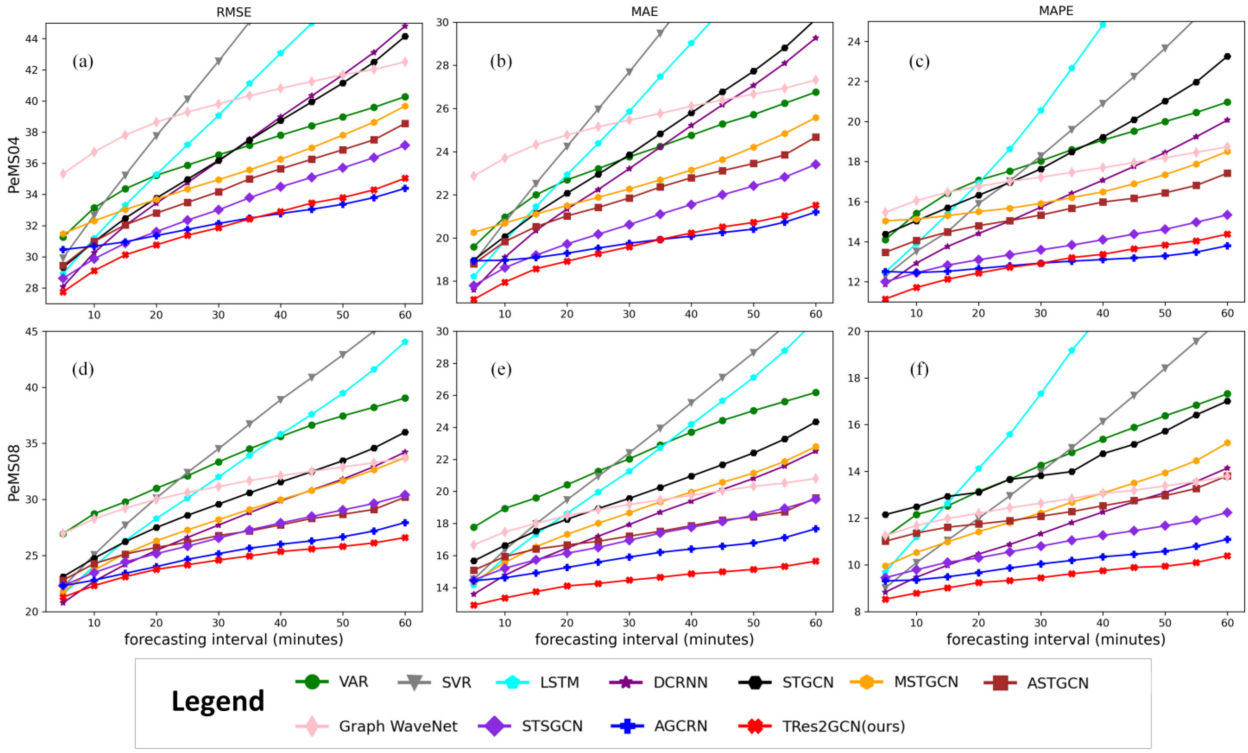

Figure 3. Model comparison under different time spans: (**a**) PeMS04-RMSE, (**b**) PeMS04-MAE, (**c**) PeMS08-MAPE, (**d**) PeMS08-RMSE, (**e**) PeMS04-MAE and (**f**) PeMS08-MAPE.

5. Discussion

5.1. Influence of Connection Method and Multi-Scale Feature

In this section, to further investigate the influence of the connection method and multi-scale change of our model, we designed seven variants of TRes2GCN and performed ablation experiments.

These were designed based on the number of spatiotemporal blocks stacked, how they were stacked and the influence of the hierarchical structure (Res2Net structure). We then compared these seven variants together with TRes2GCN on the PeMS08 dataset. During the comparison, all variant models were trained and tested with the same hyperparameters, thus ensuring fairness and scientific validity. The differences between the variant models are summarized in eight points and described below:

(1) Two Blocks-ResNet: This model is the basis of our study. It consists of two spatiotemporal blocks and is stacked in the currently most commonly used ResNet structure. Each spatiotemporal block does not contain a hierarchical structure, i.e., it contains only one TAL layer and one AGCN layer.
(2) Two Blocks-DenseNet: This model is based on the first variant with the ResNet structure replaced by the DenseNet structure.
(3) Three Blocks-ResNet: This model is based on the first variant with one more spatiotemporal block stacked and the rest unchanged.
(4) Three Blocks-DenseNet: This model is based on the third variant by replacing the ResNet structure with the DenseNet structure.
(5) Four Blocks-ResNet: This model is based on the third variant with one more spatiotemporal block stacked and the rest unchanged.
(6) Four Blocks-DenseNet: This model is based on the fifth variant by replacing the ResNet structure with the DenseNet structure.
(7) Four Blocks-DenseNet + Res2GCN: This model is based on the sixth variant with the addition of hierarchical adaptive graph convolution (Res2GCN).

(8) TRes2GCN: This is the full version of TRes2GCN. It adds hierarchical temporal attention layer on top of the seventh variant.

In the comparison of the first six variants, the enhancement of the model by the way and number of spatiotemporal blocks are explored in detail. As shown in Figure 4, we compare variants 1, 3 and 5, and we can clearly see that the effect of the way ResNet is connected is not satisfactory. When the stack is three blocks, the effect will be slightly better than two blocks; however, when the number of stacks is the fourth block, there is a significant rebound in the metrics evaluated by the model, especially the MAPE rebound is the largest. Thus, the effective number of stacked blocks for ResNet is 3, and at the fourth block, the model shows a very serious network degradation phenomenon. The occurrence of this phenomenon is most likely related to the oversmoothing of GCN [48]. It has been verified that GCNs with a stack of three or four layers will lead to a homogeneous trend of features due to the oversmoothing phenomenon, thus making the model fail. This conclusion is very similar to our experimental results. Comparing models 2, 4 and 6, we found that DenseNet has better results than ResNet in terms of connectivity. The model metrics improve as the number of spatiotemporal blocks becomes larger, and there is no network degradation. This is attributed to its high degree of feature utilization, which enriches the features while moderating the problem of oversmoothing to a certain extent. It is also very effective, as it does not increase the memory and number of parameters for the training process. However, we do not find it significantly shorter during the change of error bars. This proves that the connection method acts on the overall accuracy and does not specifically improve the accuracy at a certain time span.

It is worth mentioning that ResNet is currently used in most spatiotemporal graph neural networks; especially variant 1 (with a stacking number of 2) is very similar to the current mainstream CNN-based networks (STGCN, MSTGCN, ASTGCN, etc.). This also proves that most of the models are constructed by ignoring the influence of network deepening and connection methods on the models, and there is room for further improvement.

On the other hand, we investigated the magnitude of the improvement of our hierarchy on the model in variants 6, 7 and 8. After adding Res2GCN and Res2TAL, respectively, we find that the mean values of all three metrics gradually improve, and Res2GCN especially improves. This proves that the capture of multi-scale spatiotemporal information can, indeed, bring positive benefits to the model; the multi-scale spatial scales, especially, are more important to be considered in traffic prediction. Moreover, we also found that the error bars of variants 7 and 8 are significantly narrower compared to variant 6. This shows that we need to consider more spatial and temporal information at different scales if we want to get more accurate medium- and long-term prediction results.

Then, we further investigated the influence of the change of the Res2Net structure on the accuracy, as shown in Table 3. At the very beginning, we just applied the Res2Net optimum to our model structure, i.e., 4-scale 26 features. However, comparing the results without deploying Res2Net, the effect was improved, but only MAPE was improved more. After that, we guessed whether there were too many scales for TAL. Thus, we tried to reduce the number of scales of Res2TAL, and the effect was indeed improved. This is because TAL itself obtains the importance of each moment, and the perceptual field itself is large. The second TAL (i.e., scale equal to 3) was performed to learn the moment-to-moment association between the importance levels. Performing further TAL (i.e., scale equal to 4), on the other hand, has no practical meaning, and the results are, indeed, inferior to the case where the scale is 3. On the other hand, we also tried to reduce the number of scales of Res2GCN and found that the results did not improve, but rather had down will. Therefore, we finally chose a scale of 4 for Res2GCN and a scale of 3 for Res2TAL, both with a width of 26. This also demonstrates that the Res2Net structure can be effectively used not only in Euclidean convolution, but also in attention mechanism and non-Euclidean convolution, proving again the effectiveness and generality of its framework.

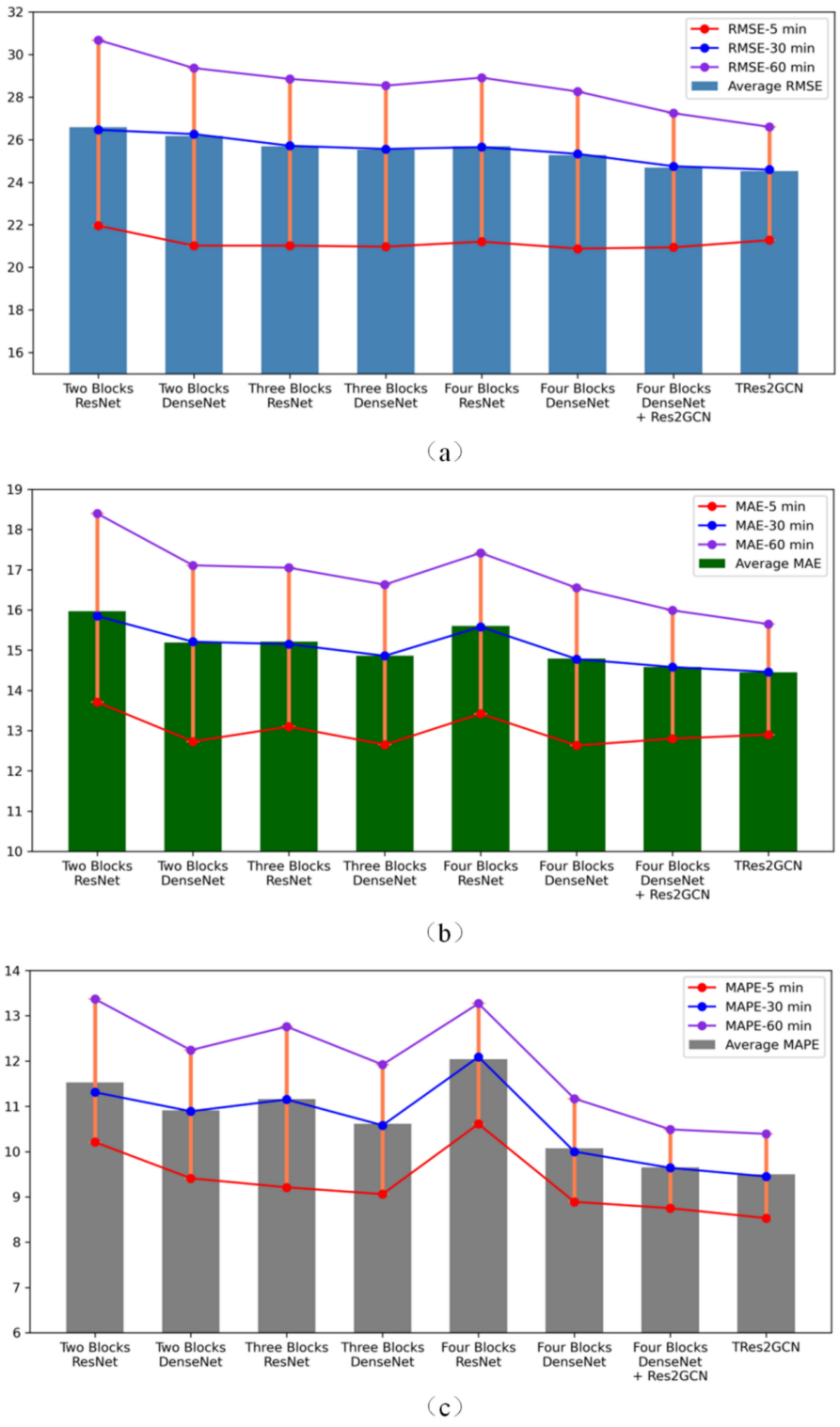

Figure 4. Visualization of the impact of components on the prediction error (PeMS08 as an example), (**a**) RMSE, (**b**) MAE and (**c**) MAPE; the error bars represent the floating changes of the predicted metrics in 60 min; the red line represents the outcome metrics predicted in 5 min; the blue line represents the outcome metrics predicted in 30 min; the purple line represents the outcome metrics predicted in 60 min.

Table 3. Influence of changes of the Res2Net structure on the model (with PeMS08 as an example).

Parameters of Res2GCN		Parameters of Res2TAL		With SE-Block	Average RMSE	Average MAE	Average MAPE (%)
Scale	Width	Scale	Width				
4	26	4	26	No	24.88	14.63	9.56
4	26	3	26	No	24.52	14.45	9.50
3	26	3	26	No	24.68	14.57	9.64
4	26	3	26	Yes	25.52	15.13	10.82

In addition, we also validate the usefulness of the SE-Block, and we find that the SE-Block causes a larger impact on the model. The main reason is that SE-Block extracts the patterns shared between scales and blurs the differences between scales. Comparing the second row with the fourth row, the results show that the Res2Net without SE-Block is more suitable for deployment in the prediction of spatiotemporal graph neural networks.

It is worth noting that since we have to cumulatively compute the results of three components (12 spatiotemporal blocks in total), and the increase in parameters per layer causes a sudden increase in the length and memory for model training. Therefore, here, we only explore the Res2Net case (4 scales of 26 widths) with no increase in computational parameters. Other settings, although they make the results better, also cause the training memory and training time of our model to plummet, so we do not take them into account.

5.2. Influence of Node Embedding Vector and Adaptive Adjacency Matrix

The node embedding vector and adaptive adjacency matrix directly affect the fine-grained spatiotemporal feature extraction, so we investigate the influence of the changes of node embedding vector and adaptive adjacency matrix in this section.

As the adaptive adjacency matrix has received attention, their methods have proliferated [29,40,49]. Inspired by these, we investigate the effect of different adjacency matrix constructions with different graph convolution embedding vectors on the prediction. However, Graph WaveNet [29] and MTGNN [49] only considered the adaptive adjacency matrix, without node embedding vector. Therefore, we adapt the composition vectors of those adaptive adjacency matrices that were not considered as vectors of node embedding and verify their suitability, as shown in Table 4.

It is worth mentioning that all these methods have been verified to simulate the adjacency matrix better, but there is no verification of the suitability of the embedding vector. Therefore, the goodness of the results of this experiment can only represent the suitability of their vectors for constructing adjacency matrices as embedding vectors for graph convolution.

Comparing the first two rows with the other rows, the adaptive adjacency matrix without adding node embedding will have similar results to the original adjacency matrix, but again, neither can obtain fine-grained node pattern variability. Moreover, no matter what embedding method is used, it will perform better than the graph convolution without embedding. This also verifies that fine-grained spatiotemporal features with node patterns are, indeed, better than ordinary spatiotemporal features.

Surprisingly, we find that the node embedding approach of AGCRN (third row) is simpler, but has better results than the other methods. In contrast, the two adaptive adjacency matrix construction methods mentioned in MTGNN are more complex (fifth and sixth rows), although it was verified in the original paper that there are better results than Graph WaveNet (fourth row without Graph Embedding) without adding node embedding. However, the results are less satisfactory when node embedding is added. This is most likely due to the overuse of fully connected layers in the building blocks, which leads to overfitting of the learned graph convolution vectors. On the other hand, comparing the third and fifth rows (undirected graphs) with the fourth and sixth rows (directed graphs) shows that the adjacency matrix learned using two vectors is generally not as good as the adjacency matrix learned using one vector. This also proves that the adaptive

undirected graph is better for prediction than the adaptive directed graph when considering node embedding.

Table 4. Influence of the adaptive adjacency matrix construction method on prediction (with PeMS08 as an example).

Method of Generating Self-Adaptive Adjacency Matrix	Node Embedding Vector	Average RMSE	Average MAE	Average MAPE (%)
Without Self-Adaptive Adjacency Matrix	Without Embedding	26.97	15.91	11.82
$ReLU(EE^T)$	Without Embedding	26.91	15.86	11.80
$ReLU(EE^T)$	$E \in R^{N \times D}$	24.52	14.45	9.50
$ReLU(E_1E_2)$	$MLP(E_1 + E_2) \in R^{N \times D}$	25.43	14.84	10.62
$ReLU(tanh(\alpha(tanh(\alpha MLP(E)) \cdot tanh(\alpha MLP(E))^T)))$	$tanh(\alpha MLP(E)) \in R^{N \times D}$	26.09	15.23	11.00
$ReLU(tanh(\alpha(tanh(\alpha MLP(E_1)) \cdot tanh(\alpha MLP(E_2))^T - \alpha(tanh(\alpha MLP(E_2)) \cdot tanh(\alpha MLP(E_1))^T)))$	$MLP(tanh(\alpha MLP(E_1)) + tanh(\alpha MLP(E_2))) \in R^{N \times D}$	26.42	15.61	11.18

5.3. Forecasting Capability over Different Spans

To visualize the prediction differences between our model and the baseline model, we randomly selected one sensor in the PeMS08 dataset to compare the prediction fit and prediction error, as shown in Figure 5.

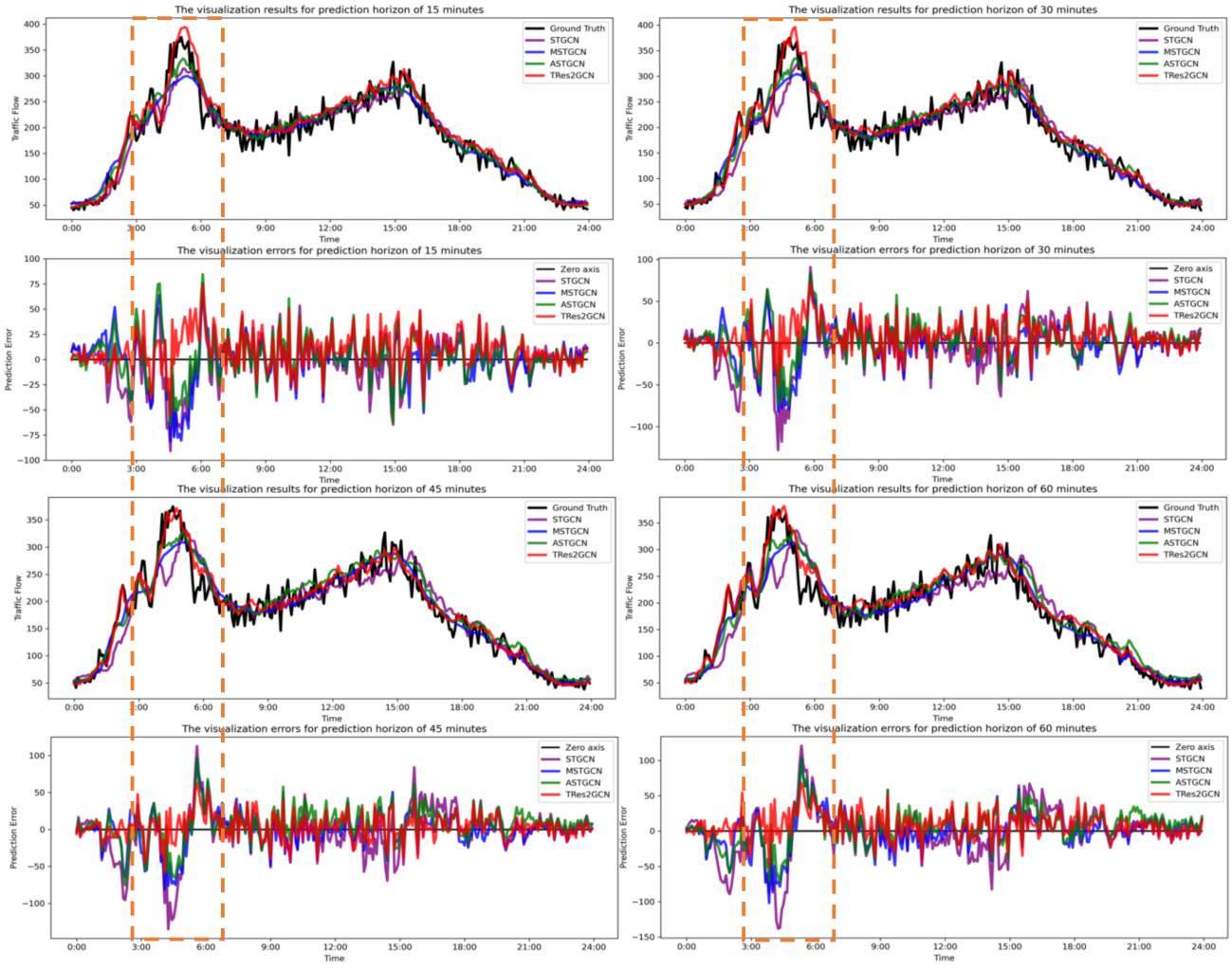

Figure 5. Visualization of prediction results and errors ranging 0–60-min span of sensor #149.

To make the comparison more specific and clearer, we visualized the prediction results under different time spans within a day. In this case, the test is performed in four time points, i.e., 15 min, 30 min, 45 min and 60 min. In each subplot, the upper plot shows the prediction volume and Ground Truth of our model with some of the baseline models, while the lower plot indicates the prediction error between the models at a distance from the Ground Truth.

The results show that the fitted curves of all three baseline models are very close for a prediction span of 15 min. And as the prediction time span increases, the error of STGCN increases significantly. In contrast, MSTGCN and ASTGCN are significantly closer to the true values than STGCN under the prediction span of 45 and 60 min. This is their effective enhancement of long-term prediction due to the consideration of multiple time periods.

Among all the models, TRes2GCN is better at predicting future traffic changes and its predicted values are closer to Ground Truth, especially in the peak part, where the advantage of our model is more prominent. We can clearly see that our model achieves better results for the part in the orange box (i.e., the peaks for all time spans). This is because the Res2Net and DenseNet structures in our model are rich in spatiotemporal features, which alleviate some of the oversmoothing problems that exist in GCN. Therefore, compared with other models, our model can not only predict the future trend of the traffic, but also predict the peak size of the traffic more accurately.

5.4. Temporal Periodic Analysis

In order to evaluate the effectiveness of fusing multiple periodic traffic features, we investigated the influence of different time periods on traffic forecasting.

By comparing rows 1, 2 and 3 in Table 5, we can see that the two models that consider daily and weekly periods respectively get different boosts compared to the model that considers only recent periods. This is because the commute usually cycles through a week. For example, the Monday pattern of the previous week will be very close to the Monday pattern of the next week. Therefore, the weekly period will be a little more elevated.

Table 5. Influence of different time periods on prediction (with PeMS08 as an example).

Time Period			With Res2Net Structure	Average RMSE	Average MAE	Average MAPE (%)
Recent Period	Daily Period	Weekly Period				
Yes	No	No	No	29.35	18.83	12.35
Yes	Yes	No	No	27.52	17.58	11.44
Yes	No	Yes	No	26.40	15.23	10.96
Yes	Yes	Yes	No	25.01	14.78	10.50
Yes	No	No	Yes	26.04	16.48	10.33
Yes	Yes	Yes	Yes	24.52	14.45	9.50

On the other hand, comparing the third row with the fourth row in Table 5, we find that even though the improvement of the day period is small, its impact must be considered if we want to further improve the prediction accuracy. This is because the daily period can be partly useful for pattern learning between weekdays (or legal holidays) during the same week. For example, traffic patterns for Monday (Saturday) can provide some contribution to the forecasting for Tuesday (Sunday).

In addition, we also explored the influence of the Res2Net structure on periodicity. Under the same conditions, we find that Res2Net improves more for models that consider only recent components. In contrast, the boost for all periods considered is smaller in comparison. This is likely due to the different sharing patterns of different period, which brings some boost but also weakens the specificity of the recent period.

5.5. Spatial Correlation Analysis

To specifically demonstrate the practical effect of the multi-scale self-adaptive adjacency matrix, we selected 15 sensors of the PeMS08 dataset for analysis. We visualized their predefined adjacency matrix, the self-adaptive adjacency matrix at a single scale and the self-adaptive adjacency matrix in the form of heat maps, as shown in Figure 6. Moreover, each row represents the degree of connection between sensors, where the more correlated nodes are the darker the color. Comparing Figure 6a with Figure 6b, Figure 6c, we see that the predefined graph is much simpler than the other two adaptive graphs. This is because the predefined graph only focuses on how the road network is connected and ignores the changes in node patterns. To validate our idea, we visualized the traffic variation of these 15 nodes on any given day and represented it as the average traffic per hour (Figure 6d–f). For example, we see that the two points that are not associated in the predefined graph, but strongly related in the traffic pattern (flow volume or variation trend) from the blue box and green box (Figure 6e,f). In the single-scale adaptive adjacency matrix, it is learned partially, but incompletely. In contrast, in the multi-scale adaptive adjacency matrix, their relationship is learned completely. This also demonstrates the need for multi-scale to be considered in the field of traffic prediction, especially the learning of the adaptive adjacency matrix.

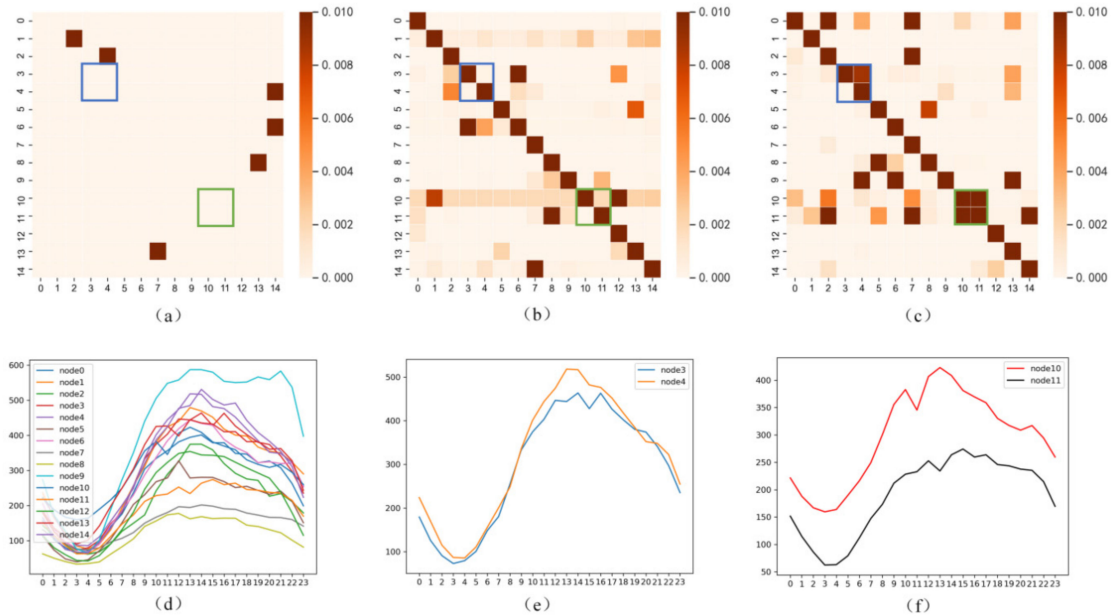

Figure 6. Part of the adjacency matrix at different scales and hourly average traffic flow variation (with PeMS08 as an example). (**a**) Predefined adjacency matrix. (**b**) Self-adaptive adjacency matrix at a single scale. (**c**) Self-adaptive adjacency matrix at multiple scales. (**d**) Hourly flow variation for the first fifteen nodes. (**e**) Hourly traffic variation of node 3 and node 4. (**f**) Hourly traffic variation of node 10 and node 11.

6. Conclusions

In this study, we propose a new STGNN model that not only considers multi-scale spatiotemporal information, but also performs adaptive learning from a fine-grained perspective. Our model is the first to use the idea of Res2Net for spatiotemporal graph neural network prediction, verifying its applicability to attention and graph convolution. We validate this model on two publicly available real-world datasets, and the results show that our model outperforms the existing state-of-the-art models. In addition, we visualized the prediction results for different traffic regions and compared them with the baseline model, clearly demonstrating the effectiveness of our proposed TRes2GCN. Unfortunately, although our model considers many characteristics of spatiotemporal data, we do not take

into account the effect of temporal data distribution. In the future, we consider the above issues while applying our method to more realistic prediction tasks.

Author Contributions: Conceptualization, Changfeng Jing; Methodology, Yi Wang; Supervision, Changfeng Jing; Writing—original draft, Yi Wang; Writing—review & editing, Changfeng Jing. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Beijing Natural Science Foundation (8222009), the National Natural Science Foundation of China (Grant # 41771412) and the Pyramid Talent Training Project of BUCEA (Grant # JDJQ20200306).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable to this article.

Acknowledgments: The authors would like to thank the anonymous reviewers for their valuable comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ahmed, M.S.; Cook, A.R. Analysis of Freeway Traffic Time-Series Data by Using Box-Jenkins Techniques. *Transp. Res. Rec.* **1979**, *722*, 1–9.
2. Chen, P.; Ding, C.; Lu, G.; Wang, Y. Short-term traffic states forecasting considering spatial–Temporal impact on an urban expressway. *Transp. Res. Rec.* **2016**, *2594*, 61–72. [CrossRef]
3. Smola, A.J.; Schölkopf, B. A tutorial on support vector regression. *Stat. Comput.* **2004**, *14*, 199–222. [CrossRef]
4. Li, H.; Liu, J.; Liu, R.W.; Xiong, N.; Wu, K.; Kim, T.-H. A dimensionality reduction-based multi-step clustering method for robust vessel trajectory analysis. *Sensors* **2017**, *17*, 1792. [CrossRef]
5. Aqib, M.; Mehmood, R.; Alzahrani, A.; Katib, I.; Albeshri, A.; Altowaijri, S.M. Smarter traffic prediction using big data, in-memory computing, deep learning and GPUs. *Sensors* **2019**, *19*, 2206. [CrossRef]
6. Bai, S.; Kolter, J.Z.; Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv*, 2018; arXiv:1803.01271.
7. Van Lint, J.; Hoogendoorn, S.; van Zuylen, H.J. Freeway travel time prediction with state-space neural networks: Modeling state-space dynamics with recurrent neural networks. *Transp. Res. Rec.* **2002**, *1811*, 30–39. [CrossRef]
8. Tian, Y.; Pan, L. Predicting short-term traffic flow by long short-term memory recurrent neural network. In Proceedings of the 2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity), Chengdu, China, 21 December 2015; pp. 153–158.
9. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. In Proceedings of the NIPS 2014 Workshop on Deep Learning, Montréal, QC, Canada, 13 December 2014.
10. Sato, R. A survey on the expressive power of graph neural networks. *arXiv*, **2020**; arXiv:2003.04078.
11. Ye, J.; Zhao, J.; Ye, K.; Xu, C. How to build a graph-based deep learning architecture in traffic domain: A survey. *IEEE Trans. Intell. Transp. Syst.* **2020**, 1–21. [CrossRef]
12. Jiang, W.; Luo, J. Graph neural network for traffic forecasting: A survey. *arXiv*, **2021**; arXiv:2101.11174.
13. Liu, J.; Guan, W. A summary of traffic flow forecasting methods. *J. Highw. Transp. Res. Dev.* **2004**, *3*, 82–85.
14. Li, W.; Wang, J.; Fan, R.; Zhang, Y.; Guo, Q.; Siddique, C.; Ban, X.J. Short-term traffic state prediction from latent structures: Accuracy vs. efficiency. *Transp. Res. Part C Emerg. Technol.* **2020**, *111*, 72–90. [CrossRef]
15. Kumar, K.; Parida, M.; Katiyar, V. Short term traffic flow prediction for a non urban highway using artificial neural network. *Procedia-Soc. Behav. Sci.* **2013**, *104*, 755–764. [CrossRef]
16. Wu, Y.; Tan, H. Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework. *arXiv*, **2016**; arXiv:1612.01022.
17. Liu, Y.; Zheng, H.; Feng, X.; Chen, Z. Short-term traffic flow prediction with Conv-LSTM. In Proceedings of the 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 11–13 October 2017; pp. 1–6.
18. Zhang, J.; Zheng, Y.; Qi, D.; Li, R.; Yi, X.; Li, T. Predicting citywide crowd flows using deep spatio-temporal residual networks. *Artif. Intell.* **2018**, *259*, 147–166. [CrossRef]
19. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv*, **2016**; arXiv:1609.02907.
20. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *Adv. Neural Inf. Processing Syst.* **2016**, *29*, 3844–3852.
21. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv*, **2017**; arXiv:1710.10903.

22. Atwood, J.; Towsley, D. Diffusion-convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 4–9 December 2016; pp. 1993–2001.
23. Zhao, L.; Song, Y.; Zhang, C.; Liu, Y.; Wang, P.; Lin, T.; Deng, M.; Li, H. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 3848–3858. [CrossRef]
24. Li, Y.; Yu, R.; Shahabi, C.; Liu, Y. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 1–3 May 2018.
25. Cui, Z.; Henrickson, K.; Ke, R.; Wang, Y. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 4883–4894. [CrossRef]
26. Guo, K.; Hu, Y.; Qian, Z.; Liu, H.; Zhang, K.; Sun, Y.; Gao, J.; Yin, B. Optimized graph convolution recurrent neural network for traffic prediction. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 1138–1149. [CrossRef]
27. Bai, J.; Zhu, J.; Song, Y.; Zhao, L.; Hou, Z.; Du, R.; Li, H. A3T-GCN: Attention temporal graph convolutional network for traffic forecasting. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 485. [CrossRef]
28. Yu, B.; Yin, H.; Zhu, Z. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; pp. 3634–3640.
29. Wu, Z.; Pan, S.; Long, G.; Jiang, J.; Zhang, C. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), Macao, China, 10–16 August 2019.
30. Guo, S.; Lin, Y.; Feng, N.; Song, C.; Wan, H. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January 2019; pp. 922–929.
31. Zheng, C.; Fan, X.; Wang, C.; Qi, J. Gman: A graph multi-attention network for traffic prediction. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 1234–1241.
32. Guo, K.; Hu, Y.; Qian, Z.; Sun, Y.; Gao, J.; Yin, B. Dynamic graph convolution network for traffic forecasting based on latent network of laplace matrix estimation. *IEEE Trans. Intell. Transp. Syst.* **2020**, 1–10. [CrossRef]
33. Hu, J.; Chen, L. Multi-Attention Based Spatial-Temporal Graph Convolution Networks for Traffic Flow Forecasting. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Online, 18–22 July 2021; pp. 1–7.
34. Liang, Y.; Zhao, Z.; Sun, L. Dynamic spatiotemporal graph convolutional neural networks for traffic data imputation with complex missing patterns. *arXiv*, **2021**; arXiv:2109.08357.
35. Bai, L.; Yao, L.; Wang, X.; Li, C.; Zhang, X. Deep spatial–temporal sequence modeling for multi-step passenger demand prediction. *Future Gener. Comput. Syst.* **2021**, *121*, 25–34. [CrossRef]
36. Zhu, J.; Wang, Q.; Tao, C.; Deng, H.; Zhao, L.; Li, H. AST-GCN: Attribute-Augmented Spatiotemporal Graph Convolutional Network for Traffic Forecasting. *IEEE Access* **2021**, *9*, 35973–35983. [CrossRef]
37. Yu, B.; Yin, H.; Zhu, Z. St-unet: A spatio-temporal u-network for graph-structured time series modeling. *arXiv*, **2019**; arXiv:1903.05631.
38. Song, C.; Lin, Y.; Guo, S.; Wan, H. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 914–921.
39. Guo, K.; Hu, Y.; Sun, Y.; Qian, S.; Gao, J.; Yin, B. Hierarchical Graph Convolution Networks for Traffic Forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, QC, Canada, 2–9 February 2021; pp. 151–159.
40. Bai, L.; Yao, L.; Li, C.; Wang, X.; Wang, C. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. *arXiv*, **2020**; arXiv:2007.02842.
41. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
42. Feng, X.; Guo, J.; Qin, B.; Liu, T.; Liu, Y. Effective deep memory networks for distant supervised relation extraction. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, Australia, 19–25 August 2017; pp. 4002–4008.
43. Gao, S.; Cheng, M.-M.; Zhao, K.; Zhang, X.-Y.; Yang, M.-H.; Torr, P.H. Res2net: A new multi-scale backbone architecture. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 652–662. [CrossRef]
44. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
45. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
46. Chen, C.; Petty, K.; Skabardonis, A.; Varaiya, P.; Jia, Z. Freeway performance measurement system: Mining loop detector data. *Transp. Res. Rec.* **2001**, *1748*, 96–102. [CrossRef]
47. Huber, P.J. Robust estimation of a location parameter. In *Breakthroughs in Statistics*; Springer: Berlin/Heidelberg, Germany, 1992; pp. 492–518.
48. Li, Q.; Han, Z.; Wu, X.-M. Deeper insights into graph convolutional networks for semi-supervised learning. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
49. Wu, Z.; Pan, S.; Long, G.; Jiang, J.; Chang, X.; Zhang, C. Connecting the dots: Multivariate time series forecasting with graph neural networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Virtual Event, CA, USA, 6–10 July 2020; pp. 753–763.